Lebanese University
Faculty of Science
Section 1

Final Exam: Machine Learning **Time: 90 minutes**

Dr. Abbas RAMMAL

Exercise 1 : Linear Perceptron

Consider the following dataset for binary classification in 3D space:

x_1	x_2	x_3	y
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

a) Write the general perceptron update rule for 3-dimensional input.
b) Initialize weights as w = [0, 0, 0] and bias = 0. Use learning rate $\eta = 1$.
Perform 2 epochs of training manually and show the updated weights and bias.
c) Discuss how increasing the dimensionality affects linear separability.

Exercise 2: Gradient Descent with Regularization

You're training a ridge regression model on a batch of two samples with the loss function:

$$L(w,b) = \frac{1}{2}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 + \lambda\|w\|_2^2$$

Given:
Samples: ($x_1 = [2, 1]$, $y_1 = 4$), ($x_2 = [1, 3]$, $y_2 = 7$)
Initial weights: w = [0, 0], b = 0
Learning rate $\eta = 0.1$, regularization $\lambda = 1$

a) Derive the gradient of the loss with respect to w and b for the batch.
b) Perform one batch gradient descent update. Show all steps.
c) Explain the trade-off between underfitting and overfitting when tuning λ.

Exercise 3: Feedforward Neural Network

You are designing a neural network for multi-class classification (3 classes):

- Input: 3 features
- Hidden Layer 1: 3 neurons (ReLU)

- Hidden Layer 2: 2 neurons (Tanh)
- Output Layer: 3 neurons (Softmax)

a) Draw the network architecture and label all layers and activations.

b) Given the input $x = [1, 0, -1]$, and simplified weights and biases:

$W_1 = [[1, 0, -1], [0, 1, 0], [1, 1, 1]]$, $b_1 = [0, 0, 0]$

$W_2 = [[1, -1, 1], [0, 2, -1]]$, $b_2 = [0, 1]$

$W_3 = [[1, 0], [-1, 1], [0, 1]]$, $b_3 = [0, 0, 0]$

Compute the output of the network (use approximations for activation functions where needed).

c) Why is Softmax preferred for the output layer in multi-class classification?

Exercise 4 : Forward and Backward Pass

You are training a simple neural network with the following setup:

- Input: $x = [x_1, x_2] = [1, 2]$
- Hidden layer activation: ReLU
- Output layer activation: Sigmoid
- Loss function: Mean Squared Error (MSE)

$$W1 = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Parameters:

  $w = [w_1, w_2] = [0.1, 0.2]$ (weights for the hidden layer)

  $b = 0.3$ (bias for the hidden layer)

  $v = 0.5$ (weight for the output layer)

  $b_{output} = -0.1$ (bias for the output layer)

  Target output: $y = 1$

  Learning rate: $\alpha = 0.01$

$$W2 = \begin{bmatrix} 1 & -1 & 1 \\ 0 & 2 & -1 \end{bmatrix}$$

$$W3 = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & 1 \end{bmatrix}$$

a) Compute the hidden layer value h

b) Compute the predicted output $\hat{y}$

c) Compute the loss using Mean Squared Error

d) Compute $\partial L/\partial v$

e) Compute $\partial L/\partial w_k$ for $w_k \in \{w_1, w_2\}$

f) Update the weights using gradient descent:

- $v \leftarrow v - \alpha * \partial L/\partial v$
- $w_k \leftarrow w_k - \alpha * \partial L/\partial w_k$

g) Explain how the gradients would change if the hidden layer activation was Tanh instead of ReLU.